

A0001. 程序员的第一课

欢迎跳进编程这个让人欢喜让人忧的大坑。今天我们来初识一下 Perl。想使用一门语言来工作, 第一步就是做好准备工作, 例如建立好程序运行的环境, 安装好所需的工具等。

如何建立程序运行的环境, 我就不再赘述了。如果已经运行过 Perl 程序, 那说明编译[compile]解释[interpret]运行环境已经 ok 了。

除此之外, 一般也会选择一种顺手的编辑工具或者 IDE (集成开发环境)。编辑工具可以辅助进行语法检查, 用多种颜色标注关键字, 帮助缩进等。然而编辑工具不是必需的, 只使用记事本等来进行编辑代码, 也是可以的。Perl 有一些有名的 IDE, 如果电脑里面已经安装了, 可以使用; 如果没有安装, 我们就不再花费更多的精力在这里, 因为它对初级应用者的帮助不是特别大, 可以等熟悉 Perl 之后再考虑。如果您还在使用记事本/写字板编辑代码, 我推荐一款简易的代码辅助编辑工具, Notepad++, 绿色小巧, 相当于升级版的记事本。业余的笔者, 就在使用 Notepad++ 编辑 Perl 代码。

准备工作就绪, 这节课就让我们来编写一个最简单的 Perl 程序, 来对 Perl 有个初步认识。

程序员都有一个不成文的 geek 习惯, 这个首个最简单的程序, 一般都是诸如在屏幕上显示 "Hello, world!" 或者 "Hello, Perl!", 用以测试自己的开发环境是否搭建完好。现在就让我们来 geek 一把!

建立一个新的程序文件, 比如 A0001.pl, 写入下面这些代码 (一定要自己敲! 不要复制):

```
1  #!/usr/bin/perl
2
3  print "Hello, Perl!";
4
5
```

运行一下试试, 是不是显示出了这句话呢~? 如果出现错误, 请仔细检查一下有没有敲错字母, 程序员可是得很细心的哦! 搞定了吧? 恭喜你! 你的第一个 Perl 程序已经成功啦!

现在我们来分析一下这个两行的程序是怎么回事。我们从第二 (三) 行 (Line3, L3) print 那句开始。

print 是我们学习的第一个函数 [function] (或者叫方法 [method]?)。print 后面接一个双引号, 引号之间填入想在屏幕上显示的内容。另外要注意每句代码之后都要加一个分号表示这句结束。

我们再来试试这个代码:

```
1  #!/usr/bin/perl
2
3  print "Hello, Perl!";
4  print "Hello, world!";
```

运行一下, 会发现, 两个短句连在一起输出到屏幕上了对吧! 这也就是说, print 不会自动添加分隔符 (比如空几格或者换行)。在实际应用中, 我们往往想让它另起一行显示, 那我们应该怎么写呢?

我们如下改写上述代码:

```
1  #!/usr/bin/perl
2
3  print "Hello, Perl!\n";
4  print "Hello, world!";
```

注意,我在第一句之后加了一个\n.运行一下,发现换行了吧!这个\n,就是换行符.如果你想在打印到某个地方的时候另起一行,只需加个\n 就行啦.现在我们可以试试把\n 加到任何别的地方,比如两个词之间,等.

看下面这个代码,和上面的代码是等效的,体会一下?应该不难理解吧.

```
1  #!/usr/bin/perl
2
3  print "Hello, Perl!\nHello, world!";
```

你可能会觉得\n 这样的字符很奇怪,也会有一些疑问,如有兴趣可以参考各类教材中的<转义字符>相关章节,没兴趣会用\n 就行了.

print 会将引号中的东西几乎原样输出,引号中扩着几个空格,屏幕上就有几个空格.另外输出是依次,程序不会私自添加空格或者换行,除非程序员添加\n.斜杠方向别反了.由此也有一个推论,每次 print 时最后加个\n 也算是某种好习惯(当然是需要的时候)?

现在我们回过头来看第一行(L1)程序.Pperl 中,以#开头的行,都是注释.注释的意思是,这行就是个草稿,执行的时候会忽视这行.注释的内容随便写,不会影响程序执行.一般都是程序员做个标记,方便自己阅读程序使用的.我来给出一些例子,阅读体会一下:

```
1  #!/usr/bin/perl
2
3  # Code Example of A0001, First day of a geek programmer
4  # Author: Ting Chen, Chris
5
6  print "Hello, Perl!\nHello, world!";
```

这里,我们用注释的方法标记了一些代码信息.Notepad++自动识别注释部分并且用绿色显示.

```
1  #!/usr/bin/perl
2
3  print "Hello\nworld"; #\n means move to next line
```

这里,我们用注释来标注一下对这行的理解.

```

1  #!/usr/bin/perl
2
3  #*****#
4  #                                           #
5  #           Class A0001 Example Code        #
6  #                                           #
7  #*****#
8
9
10 print "Hello\nworld";

```

这里,我们画的很像个图形,但仔细观察,每行开头都是个#,所以这些其实是个注释,是不是很聪明的想法呢?

在程序中添加合适的关键的注释,也是一个好习惯,方便别人看,也方便自己改.

这个程序的第一行,其实是个比较特别的注释,或者说,这句是个特例,并不是毫无意义.它代表着在 linux 系统下,perl 所在的位置(就是你把 perl 安装到哪儿了).不理解也没关系,每次写程序之前,把这句抄上去放在最前面就行了.L1 这个路径不能随意改变,而别的以#开头的注释却完全无所谓,随便怎么写都行.重申一下,仅此一句(L1)是个特例.如果有兴趣了解更多关于注释的语法,可以参考各类教材或者上网搜一下.

学习写程序呢,最主要的是多敲,就是说,看到什么代码都要自己敲下来,跑一跑,改一改,再跑一跑,多尝试,多测试.特别是当你想“如果我写成 xxx 样子,那会怎么样?”的时候,一般最好是自己写一写跑一跑,就能明白很多了.

这节课提到的东西,都要自己敲下来,跑跑试试,一共也没几个字母,不会太繁琐的.如果你觉得为了上这节课做了很多繁琐的准备工作,比如安装软件,设置环境,那大可不必让它影响到你学习的兴趣,因为这都是一次性工作,不会再有.

这节课基本上就是先对程序有个初步的认识,如何运用变量,如何写不同的控制结构,如何运用到实际的数据处理中,我们会马上开始,不要急哦~!

思考题:把这个显示到屏幕上(一共 5 行):

```

*****
*           *
*  Haha! I'm gonna be a coder!  *
*           *
*****

```